04

Ejemplos adicionales usando equals o hashCode

Transcripción

[00:00] Hola. ¿Cómo están? Vamos a iniciar nuestra clase número 15. Nuestra clase 15 es la última del bloque métodos equals y hashCode. Aquí vamos a ver unos ejemplos adicionales y también vamos a recapitular lo que hemos visto en la clase 14, que eso es muy importante. Por ejemplo en nuestra clase 14 vimos que la diferencia en una lista termina siendo, prácticamente tenemos unas posiciones.

[00:28] En nuestro hashSet por ejemplo o cualquier tipo de lista hash no tenemos una posición fija. Entonces ahí vimos lo siguiente, que si tenemos bastantes objetos creados, para obtener un objeto se va a hacer bien difícil en toda mi red prácticamente en el mundo Java. ¿Entonces qué es lo que hace el hash? Simplemente él va a agrupando en una pequeña cajita alumnos objetos.

[01:12] Él agrupa aquí y va agrupando por ejemplo dependiendo del hash que nosotros estamos colocando. Por ejemplo en esta situación nosotros colocamos un hash, ¿de qué tipo? De tipo nombre, entonces aquí va agrupando por ejemplo los nombres. Vamos a ver aquí, lo colocamos aquí frontal, y aquí vamos a colocarlo atrás. Aquí vamos a hacer clic, atrás listo.

[01:57] Y aquí por ejemplo hemos agrupado en una pequeña cajita. Así cuando nosotros buscamos por un hash ya sabemos en cuál cajita buscar y ya sabemos cuál es el objeto que estamos necesitando. Esa es una parte importante. También el otro punto importante es ver que en la parte de lo que es el Alumno también utilizar siempre, si utilizamos el equals, también utilizar el hashCode sobrescribir estos métodos.

[02:28] ¿Por qué? Porque va a haber situaciones que por ejemplo, en la clase 14 nosotros no sabíamos si utilizar tal vez ArrayList o tal vez LinkedList o tal vez HashSet. Si no sabemos, ¿qué sucede? Puede ser que de algún momento a otro cuando cambiamos de lista para hash, ahí va a dar un problema.

[02:49] Tal vez cuando queramos ver un equals o cuando queramos utilizar un contains, es un punto muy importante. Para esto, en nuestra clase 15 vamos a duplicar nuestra clase 14, quitamos clase 15 en nuestro curso, vamos ahora a hacer la parte de matrícula, colocar nuestros alumnos en un curso X. List. Vamos a colocar Collection alumnos = new HashSet. Listo.

[03:34] Vamos ahora a crear nuestro método de adicionar, addAlumno. Aquí tenemos dos alumnos y utilizamos aquí lo siguiente: {this.alumnos.add(alumno);}. Una vez hecho esto, después tenemos aquí el public boolean para ver si este alumno ya existe, que estamos adicionando. Vamos a ver: verificaAlumno. Aquí colocamos Alumno alumno y colocamos {this.alumnos.contains(alumno);}. Perfecto.

[04:28] Aquí damos un return, que está retornando un booleano, el contains. En nuestra clase 15 tenemos ya la lista de alumnos. Vamos a crear ahora nuestra lista de cursos. Vamos a ver por ejemplo nuestra clase 12. Aquí tenemos nuestra clase 10, vamos a ver si aquí tenemos una lista de cursos, por ejemplo, vamos a colocar el curso 1.

[05:05] Curso 1. En nuestro curso 1 ahora ya adicionamos la lista, ahora hacemos lo siguiente: curso1.addAlumno y adicionamos todos nuestros alumnos a este curso. Estamos viendo tipo un escenario que puede ser en la vida real. Tenemos las clases y tenemos los alumnos registrados a las clases y al curso.

[05:51] Aquí tenemos alumno 1, 2, 3, 4, 5, 6, 7, perfecto. Aquí tenemos alumnoNuevo que también estamos comparado y aquí por ejemplo vamos a hacer los mismo. Punto. Aquí no estamos implementando. ¿Nos faltó aquí qué

cosa? Implementar nuestro método get.Alumnos. Vamos a utilizar aquí Generate getters y vamos a colocar aquí alumnos, .getAlumnos.

[06:27] Y aquí usamos .getAlumnos y listo. Una vez hecho esto, ejecutamos nuevamente y true true. Prácticamente lo que estuvimos haciendo aquí ¿qué cosa fue? Fue prácticamente externalizar, dejar un poco más responsabilidad a nuestra clase curso, prácticamente, para que él tenga todo lo que sea relacionado a alumno, aula, tal vez matrícula, lo que sea necesario.

[07:11] Y aquí en nuestros ejemplos utilizar prácticamente lo que necesitamos, que sería adicionar. Ya no necesitamos aquí adicionar porque ya tenemos el método adicionar y eso facilita también, ¿qué cosa? Que podemos cambiar los tipos que nosotros queramos y solamente ejecutamos en un solo lado y prácticamente va a afectar a todas nuestras clases que utilizan ese tipo de implementación.

[07:35] Esa parte es muy importante siempre usarla. No dejar que muchas cosas de lógica estén dentro prácticamente de alguna regla, sino que prácticamente todo esté dentro de nuestras clases que estamos creando. Y en la parte de ejecución simplemente tengamos ya los métodos ya para usar.

[07:58] Tenemos el método addAlumno, también podemos crear el método de remover alumno. Podemos ver el método de verificar alumno, por ejemplo aquí. Aquí estaríamos utilizando el contains. ¿Por qué? Porque ahora tenemos el curso.verificaAlumno y aquí mandamos el alumno. Y él ya está recibiendo la lista, ya tiene la lista de alumnos y daba true.

[08:23] Ya no necesitamos hacer contanis y todo eso. ¿Por qué? Porque todo esto de ahí ya está dentro de nuestra clase curso. Es muy bueno tener presente esa parte ahí, es bien interesante. Si bien es un poco también difícil o al inicio tal vez entender por qué utilizar el hashSet con el equals, pero espero que en estos ejemplos que también están aquí, tengan ya una noción y para probar jugando aquí un poco.

[08:57] Por ejemplo aquí ya no quiero que sea, aquí ya sabemos que el alumno, entonces alumno le voy a quitar aquí el hashSet, le voy a quitar el hashCode. Lo quito y ejecuto como estuvimos haciendo acá. Ah, va a dar un false. Claro, como las cajitas, él no sabe en cuál cajita buscar. Por aquí ya sabe que el código va a buscar por código.

[09:22] Aquí tenemos que tener el nombre, en realidad. ¿Por qué? Porque recuerden que habíamos visto, habíamos dicho ¿qué cosa? Si utilizamos el equals por nombre también el hashCode también tiene que ser por nombre. Para que sea haga único, para que sea más fácil tal vez la parte de mantener el código. Esto sería todo en nuestra clase número 15, estamos terminando el bloque métodos equals y hashCode, y nos vemos en nuestra siguiente clase. Muchas gracias.